VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

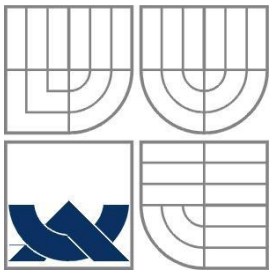
MOBILNÍ APLIKACE PRO SKENOVÁNÍ SÍTĚ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

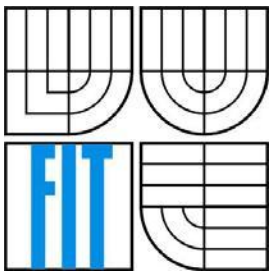
AUTOR PRÁCE
AUTHOR

MAREK TEUCHNER

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MOBILNÍ APLIKACE PRO SKENOVÁNÍ SÍTĚ

MOBILE APPLICATION FOR NETWORK SCANNING

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — MAREK TEUCHNER
AUTHOR

VEDOUCÍ PRÁCE — Ing. MICHAL KOVÁČIK
SUPERVISOR

BRNO 2015

Abstrakt

Předmětem této bakalářské práce je navrhnout a implementovat mobilní aplikaci pro operační systém Android, která umožňuje nalézt připojená zařízení v síti, zobrazit detaily o síti a jednotlivých připojených zařízeních. Pro každé nalezené zařízení lze vybrat vlastní ikonu a přidat poznámku. Aplikace také zaznamenává logy a statistiky. Vybrané zařízení lze také napadnout pomocí útoků typu Denial of Service.

Abstract

The subject of this bachelor's thesis is to design and implement an application for Android operating system that allows the discovery of all connected devices in a local network and to display information about it and all of its devices. The application would allow the user to assign a custom icon and a note to each device within the network. Logs and statistics are also collected by the application. In addition, it is possible to attack the selected device using various Denial of Service attacks.

Klíčová slova

Skenování sítě, síťové útoky, analýza zařízení v síti, DoS útoky

Keywords

Network scanning, network attacks, network device analysis, DoS attacks

Citace

Marek Teuchner: Mobilní aplikace pro skenování sítě, bakalářská práce, Brno, FIT VUT v Brně, 2015

Mobilní aplikace pro skenování sítě

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana inženýra Michala Kováčika.

Marek Teuchner
19. května 2015

Poděkování

Rád bych poděkoval svému vedoucímu práce Ing. Michalu Kováčikovi za rady a připomínky při vývoji aplikace.

Obsah

1. Úvod

V dnešní době si už málokdo dokáže představit svět bez mobilního telefonu. Telefony jsou již dneska na tak vyspělé úrovni, že dokážou plnohodnotně nahradit počítače. Z toho důvodu je vhodné mít v nich k také odpovídající software, kdekoliv a kdykoliv, okamžitě k dispozici. Jedním z druhů pokročilých mobilních aplikací jsou nástroje pro skenování sítě.

Tyto nástroje jsou nezbytným vybavením každého správce sítě, stále častěji je však vyhledávají také běžní uživatelné. Skenery sítě dávají uživatelům úplný přehled o tom, kdo společně s nimi síť využívá. Některé sofistikovanější nástroje dokážou zjistit velmi detailní informace o každém připojeném zařízení. Na poli mobilních aplikací moc kvalitních nástrojů ještě neexistuje, většina podobných aplikací totiž nedokáže analyzovat zařízení do hloubky nebo provádět síťové útoky.

Poměrně horkou novinkou na poli mobilních aplikací jsou síťové útoky. Aplikace, které je dokážou interpretovat, však mají uživatelům sloužit k užitku a jsou nezbytným pomocníkem při odhalování bezpečnostních anomálií v síti.

V kapitole 2 uvádím veškeré teoretické informace ohledně principů skenování sítě a portů a také informace o tom, jak útoky, které aplikace dokáže provést, fungují. Ve třetí kapitole popisuji návrh aplikace a v následující kapitole její implementaci. Pátá kapitola pojednává o testování aplikace na reálných sítích a v poslední kapitole uvádím shrnutí celé práce.

2. Principy skenování sítě a útoků typu Denial of Service

V této kapitole rozeberu jednotlivé způsoby nalezení zařízení v síti, objevení otevřených portů na daném zařízení a také několik síťových útoků, které mají za následek omezení nebo dokonce úplný výpadek služby (z angličtiny tzv. *Denial of Service*).

2.1. Skenování sítě

Princip skenování sítě spočívá ve hromadném odeslání paketů na všechny IP adresy, u kterých chceme zjistit přítomnost zařízení. Existuje několik různých způsobů, jak lze dané zařízení objevit.

2.1.1. Zprávy ICMP

Tento princip skenování sítě je nejjednodušší. Na cílovou IP adresu se odešle zpráva ICMP a pokud se vrátí odpověď, je zařízení připojeno. Některé zařízení však mohou tyto zprávy záměrně blokovat, proto je pro dosažení úspěšného výsledku potřeba použít i jiné metody. [1]

2.1.2. TCP SYN

Výhoda použití této možnosti je ta, že nedochází k uzavření žádného spojení a cílové zařízení se ani nedoví o pokusu o skenování. Tento princip se také jinak nazývá *TCP Stealth Scan*. Dojde k odeslání prázdného TCP paketu s nastaveným příznakem SYN. Tím je naznačen pokus o uzavření spojení. V případě neúspěchu je spojení odmítnuto a je vrácen paket s příznakem RST (reset). Pokud se připojení podaří, je vrácen paket s příznakem SYN/ACK. Namísto potvrzení spojení (paket ACK) je však odeslán paket RST a tím je pokus o navázání spojení přerušen. [1]

2.1.3. TCP ACK

Technika této metody je velmi podobná technice TCP SYN. S tím rozdílem, že namísto SYN paketu je odeslán paket ACK. Tento paket slouží k signalizaci úspěšně navázaného spojení. Jelikož ale žádné spojení navázáno není, druhá strana odpoví paketem RST. Pokud však nepřijde žádná reakce, sken je neúspěšný. [1]

2.1.4. UDP Ping

Jedná se skenovací metodu odesílající paket pomocí protokolu UDP. V tomto případě je žádoucí posílat paket na pravděpodobně zavřený port. Pokud je na dané IP adrese přítomno nějaké zařízení a zvolený port je uzavřen, odešle zpět paket *ICMP Port Unreachable*. V opačném případě zůstane paket bez odpovědi. Použití této metody je vhodné, pokud dané zařízení disponuje filtrem na TCP porty. [1]

2.1.5. SCTP INIT Ping

Podstatou této metody je odeslání INIT paketu pomocí protokolu SCTP na zvolený port. V případě zavřeného portu se jako odpověď vrátí paket ABORT. Pokud je port otevřený, vrátí se paket s příznakem další fáze uzavírání spojení – INIT-ACK. Na to mu je však odpovězeno paketem ABORT, který pokus o uzavření spojení ukončí. Každá z těchto odpovědí signalizuje, že zařízení je k dispozici. [1]

2.1.6. IP Protocol Ping

Jednou z novějších skenovacích technik je IP Protocol ping. Pomocí této techniky lze zařízení skenovat pomocí různých protokolů. Je třeba odeslat paket s korektní hlavičkou pro daný protokol. Odpověď pomocí stejného protokolu nebo paket *ICMP Protocol Unreachable* signalizuje připojené zařízení. [1]

2.1.7. ARP Ping

ARP (*Address Resolution Protocol*) je protokol, který slouží k překladu IP adresy na MAC adresu. Zmíněná metoda využívá tohoto protokolu. Dojde k odeslání ARP dotazu pro danou IP adresu na *broadcast* adresu sítě. V případě, že se vrátí ARP odpověď, je zařízení dostupné.

2.2. Skenování portů

Některé z výše popsaných metod lze použít také pro skenování portů. Následuje seznam dalších metod zaměřených přímo na zjištění otevřených portů.

2.2.1. TCP Connect

Jedná se o základní a málo efektivní metodu. Oproti ostatním je zdlouhavá, málo úspěšná a navíc se cíl dozví o provedeném skenu díky logování. Probíhá pomocí klasického připojení přes BSD schránky. V případě odpovědi je zařízení aktivní. [1]

2.2.2. Metody TCP NULL, FIN, Xmas a Maimon

Tyto speciální metody se liší od ostatních TCP metod tím, že neposílají SYN, ACK ani RST paket. RFC 793 [2] uvádí, že se v případě zavřeného portu vrací jako odpověď paket RST. V opačném případě zůstane dotaz bez odpovědi. TCP NULL posílá prázdný paket (příznaky TCP jsou nastaveny na nulu), TCP FIN posílá pouze FIN paket, TCP XMAS odesílá paket s příznaky FIN, PSH a URG a TCP Maimon paket FIN/ACK. [1]

2.2.3. TCP Window

Tato metoda je velmi podobná metodě TCP ACK s tím rozdílem, že navíc dochází k analýze vráceného RST paketu. Velikost okna paketu pak udává stav portu. Pokud je okno prázdné, jedná se o zavřený port, pokud okno obsahuje nějaká data, znamená to otevřený port. [1]

2.3. Síťové útoky

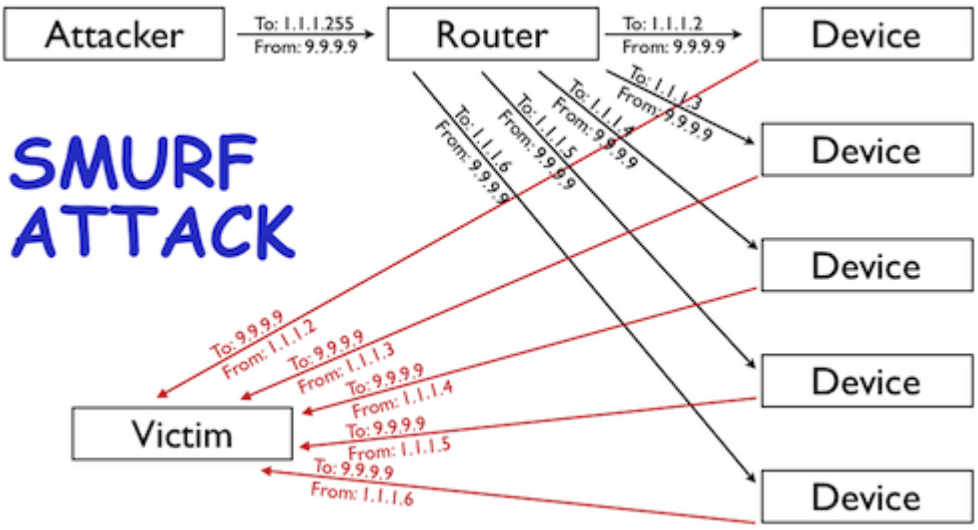
Každá síť by měla být dobře zabezpečena proti útokům, aby byli její uživatelé v bezpečí před možnými hrozbami. Aby se však dalo útokům bránit, je třeba nejprve pochopit jejich princip. Uvádím zde seznam nejrozšířenějších útoků, jejich podstatu a také způsoby obrany vůči nim.

2.3.1. ICMP Echo flood

ICMP Echo flood neboli *Ping flood* je jednoduchý DoS (*Denial of Service*, odmítnutí služby) útok, jehož podstata spočívá v zavalení oběti velkým množstvím paketů ICMP s kódem 8 (*echo request*). Oběť je nucena odpovídat na tyto požadavky a tím dojde k zahlcení šířky pásma pro odchozí i příchozí data. Tento útok je znám už delší dobu, přesto je stále efektivní. Účinnou obranou je zablokování příchozích požadavků z dané IP adresy. [3]

2.3.2. Smurf

Jedná se o reflektivní ICMP útok. Útočník pošle velké množství ICMP Echo dotazů s podvrženou zdrojovou IP adresou (vydává se za oběť) na *broadcast* adresu sítě. Směrovač pošle přes každé zařízení odpověď oběti. Čím více zařízení je přítomno v síti, tím silnější je útok. Obrana je poměrně náročná – je potřeba na směrovači nastavit správnou identifikaci ICMP žádostí. [4]



Obrázek 2.1: Grafické znázornění Smurf útoku [9]

2.3.3. Fraggle

Tato variace útoku Smurf posílá místo paketů ICMP pakety UDP na port 7 nebo 19 (Implementována verze posílající pakety na port 7). Efekt je obdobný. Jedná se o zastaralý útok, jelikož dnešní zařízení nemají zpravidla port 7 otevřen. Nicméně v některých případech může být stále účinný. [4]

2.3.4. Smack

Jedná se o variaci Ping flood útoku, která odesílá pakety ICMP 3 (Unreachable) z podvržených náhodných IP adres. Obrana je téměř nemožná. [5]

2.3.5. Twinge

Cílem tohoto útoku je zahltit oběť pakety typu ICMP s náhodným kódem. Pakety mají navíc podvrženou náhodnou IP adresu. Obranou může být například zahazování ping žádostí z adres mimo místní síť. [5]

2.3.6. P-smash

Jedná se variaci Ping Flood útoku, která posílá pakety typu ICMP 9 (*Router Advertisement*) z podvržených náhodných IP adres. Tento útok je v dnešní době již na mnoha zařízení opraven. [5] [6]

2.3.7. WinFreez

Tento experimentální útok byl vyvinut ke zpomalení zařízení s operačním systémem Windows. Principem je zavalení oběti pakety typu ICMP Redirect s adresou přesměrování do sebe sama. Útočník se vydává za směrovač. Tento útok je již dnes pro novější verze Windows pravděpodobně opraven, nicméně může fungovat pro starší verze (např. Windows XP) [5]

2.3.8. UDP Zero flood

Další ze záplavových útoků, tentokrát se pakety odesílají prostřednictvím protokolu UDP na port 0. Lze se proti němu bránit zahazováním paketů přicházející na nulový port. [5]

2.3.9. ARP Cache Poisoning

Tento druh útoku funguje pouze na místní síti, je ale maximálně účinný i v dnešní době. Přesněji se jedná o útok *Man in the middle*. Tímto útokem dokáže útočník absolutně odříznout oběť od přístupu k internetu. Principem útoku je napadení a modifikace *ARP cache* – jedná se o lokální databázi každého zařízení v síti obsahující informace o tom, které IP adrese patří která MAC adresa. Jednotlivá zařízení v síti mezi sebou pravidelně komunikují pomocí ARP dotazů a na základě obdržených odpovědí udržují *ARP cache* aktuální.

Útočník vytvoří vlastní podvržené odpovědi, které pošle dvěma zařízením v síti (nejčastěji směrovač a oběť) a „postaví se“ mezi ně. Veškerá síťová komunikace mezi těmito dvěma zařízeními pak bude procházet skrze útočníka. Útočník tyto pakety poté jednoduše zahodí. [7]

Tento útok lze nejlépe demonstrovat příkladem. Mějme směrovač A, útočníka B a oběť C. Útočník si připraví dva druhy paketů s následujícími parametry:

Typ paketu:	ARP Reply
Zdrojová IP:	A
Zdrojová MAC:	B
Cílová IP:	C
Cílová MAC:	C

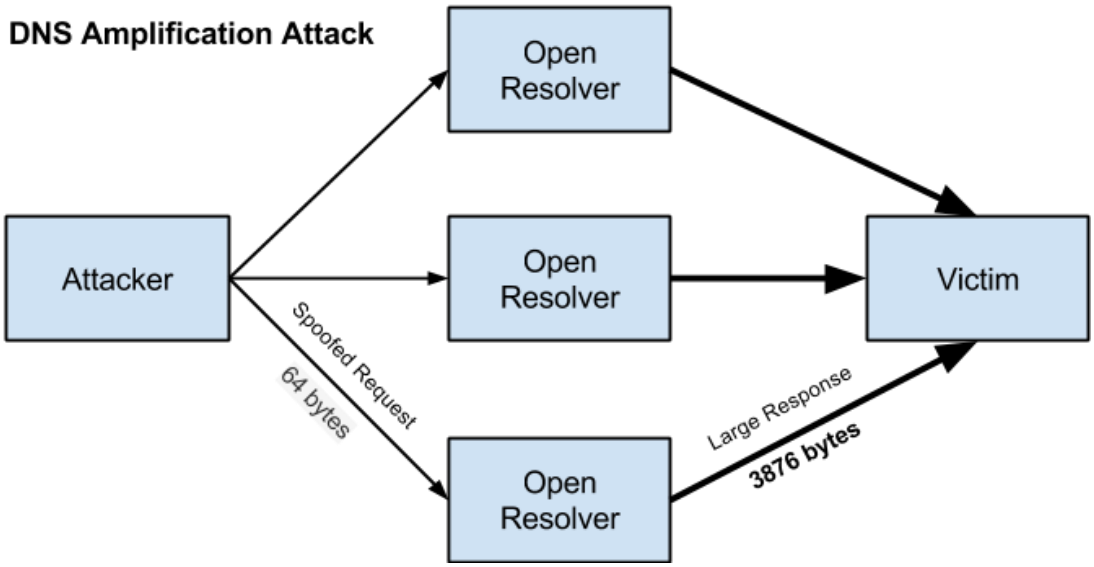
Typ paketu:	ARP Reply
Zdrojová IP:	C
Zdrojová MAC:	B
Cílová IP:	A
Cílová MAC:	A

Tyto pakety jsou pak v pravidelných intervalech odesílány na příslušná zařízení.

2.3.10. DNS Amplification Attack

Jedná se o jeden z nejnebezpečnějších a nejvýkonnějších útoků dneška. Jeho hlavní výhoda je ta, že dokáže dosáhnout velmi vysokého koeficientu amplifikace – teoretické maximum je přes 80. Jako prostředek útoku využívá DNS server. Útok je nejčastěji prováděn přes místní DNS server díky nízké latenci.

Útok spočívá v odesílání upravených DNS paketů s podvrženou zdrojovou IP adresou jako oběť na DNS server. Podvržen je rovněž zdrojový port, který je generován náhodně. Cílovým portem je UDP port 53. Oběti poté přijde odpověď, která je několikanásobně větší, než je velikost odeslaného dotazu. Maximální možná velikost odpovědi může být až 4096B. Dotaz má velikost pouze přibližně 60B. Pro dosažení co největší amplifikace je vhodné použít doménu s rozsáhlým DNS záznamem. [10]



Obrázek 10.2: Schéma DNS Amplification útoku [11]

Formát datové části DNS dotazu vypadá následovně [8]:

XXXX – Náhodné ID transakce (hex.)
0100 – příznaky – jedná se o DNS dotaz, odpověď nebude oříznuta na 512B, rekurzivní překlad je požadován.
0001 – následuje jedna otázka
0000 – následuje nula odpovědí
0000 – následuje nula záznamů
0000 – žádná další data

Nyní je potřeba rozdělit doménu na jednotlivé části. Například doména *google.com* bude rozdělena na části *google* o délce 6 a *com* o délce 3. Tečky oddělující jednotlivé části jsou ignorovány.
XX – počet bytů první části v hexadecimálním tvaru. Maximálně 256B.
XXXX... – název první části převedený do hexadecimálního tvaru (např. znak ‚*o*' je reprezentován ASCII hodnotou 111 a jeho podoba v hex. tvaru je 6F)
YY – počet bytů druhé části
YYYY... – název druhé části atd.

00 – symbolizuje ukončení jména domény
00FF – typ záznamu: ANY – dotaz na všechny dostupné typy záznamu [12]
00FF – třída záznamu: ANY – dotaz na všechny dostupné třídy záznamu [12]

Výsledná datová část paketu například pro doménu *google.com* má následující tvar:

```
12 34 01 00 00 01 00 00 00 00 00 00 03 77 77 77 06 67 6F 6F 67
6C 65 00 00 FF 00 FF
```

Proti tomuto útoku se dá bránit například vhodným nastavením *Response Rate Limiting* na použitém DNS serveru [10]. Tím dojde k omezení frekvence odesílání jednotlivých odpovědí.

3. Návrh aplikace

Tato kapitola pojednává o návrhu aplikace, její funkčnosti a také grafického uživatelského rozhraní. Aplikaci jsem se rozhodl pojmenovat *Network Beyond*, přeloženo jako „Za sítí“. Název má charakterizovat nové možnosti při skenování sítě a také to, že s touto aplikací lze provádět něco víc, než je zatím u konkurenčních aplikací možné. Celá aplikace je v anglickém jazyce.

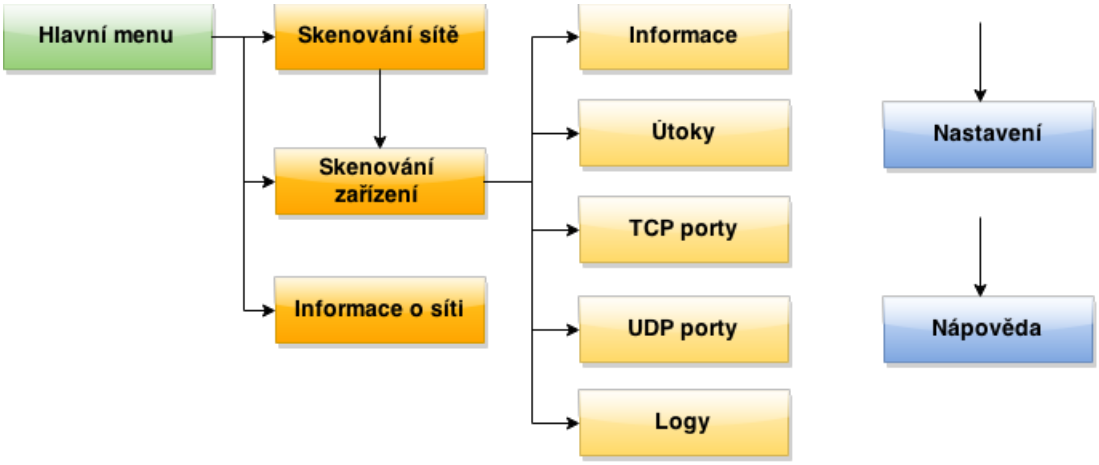
3.1. Požadovaná funkčnost aplikace

Aplikace musí umět provádět následující:

- Skenovat celou síť a přehledně zobrazit všechna nalezená zařízení
- O každém zařízení zobrazit jeho detaily, tzn. IP adresu, MAC adresu, jméno výrobce síťové karty zařízení, síťové jméno, NetBios jméno a otevřené TCP a UDP porty
- Na vybrané zařízení umožnit provést síťové útoky typu Denial of Service
- Ke každému zařízení přiřadit vlastní ikonu a poznámku
- O každém zařízení uchovávat statistiky – datum a čas nalezení, externí IP adresu a také jméno přístupového bodu.
- Zobrazit veškeré informace o síti

3.2. Logická struktura aplikace

Jelikož je aplikace poměrně rozsáhlá, bylo nejprve potřeba vymezit jisté logické části aplikace. Po načtení aplikace lze z hlavního menu provést skenování sítě, skenování vybrané IP adresy, která může být interní nebo externí, nebo zobrazit informace o síti. Ze všech částí aplikace lze měnit nastavení nebo zobrazit nápovědu. Po úspěšném skenování sítě lze dále přistupovat k jednotlivým zařízením, zobrazit informace o nich a také zaútočit. Úplná logická struktura je vyobrazena v grafu 3.1.



Obrázek 3.1: Logická struktura aplikace. Položky Nastavení a Nápověda jsou přístupné ze všech částí aplikace.

3.3. Po spuštění

Po spuštění aplikace bude uživatel dotázán na povolení *Superuser* neboli *root* práv. Superuser práva jsou nezbytnou součástí pro plnou funkčnost aplikace a bez tohoto povolení nemůže aplikace fungovat – vypíše chybovou hlášku a po jejím potvrzení se aplikace ukončí. Po potvrzení se zobrazí načítací dialog, během kterého se aplikace inicializuje.

Aplikace nejprve zaregistruje *broadcast receiver*, který automaticky hlídá stav připojení k síti. Pokud dojde k uzavření nového spojení, aplikace automaticky načte nové síťové informace. Pokud se spojení přeruší, aplikace ukončí všechny stávající činnosti a nepovolí další, dokud nebude připojení znovu obnoveno.

Dále si aplikace načte externí knihovny použité pro skenování a připraví je k použití. Poté načte veškerá uložená nastavení, ikony, obrázky a připraví databázi. Pokud se aplikace spouští poprvé, dojde k sestavení databáze portů z externího souboru.

Po úspěšné inicializaci se zobrazí hlavní menu aplikace.

3.4. Hlaví menu

Hlavní nabídka aplikace je navržená velmi jednoduše. Obsahuje logo aplikace a dále tři tlačítka – *Scan Network* (skenování sítě), *Target IP* (ruční zadání IP adresy pro analýzu) a *Display Network Info* (zobrazení informací o síti).

3.5. Skenování sítě

Po kliknutí na *Scan Network* se zobrazí nabídka s výpisem z posledního skenování (v případě nového skenování prázdný seznam), tlačítkem sloužícím k ovládání skenování a textem zobrazujícím aktuální stav skenování.

3.5.1. Skenování

Po spuštění skenování (tlačítko *Start Scan*) začne aplikace postupně interpretovat zvolené skenovací metody (více v kapitole 3.5.2.) a po jejich dokončení aplikace vyhledá v databázi ikony a poznámky pro jednotlivá zařízení a zobrazí je. Aplikace automaticky identifikuje výchozí bránu sítě (přiřadí mu ikonu směrovače a poznámku *# Gateway #* (= brána)) a také zařízení, na kterém je aplikace spuštěna (ikona zeleného trojúhelníku a poznámka *# YOU #*). V případě nastavení vlastní ikony nebo poznámky pro dané zařízení budou zobrazeny vlastní údaje. Dále aplikace zjistí NetBios jména zařízení, pokud je tato možnost vybrána v nastavení.

Výsledkem skenování je výpis jednotlivých zařízení v seznamu. Pro každé zařízení je uvedena IP a MAC adresa, poznámka a ikona. V případě neznámého zařízení není poznámka vyplněna a jeho ikonou je bílý otazník v modrém kruhu. Po kliknutí na vybrané zařízení se spustí jeho analýza.

Skenování lze také úplně přerušit stisknutím ovládacího tlačítka. Tento proces může trvat až několik sekund, zvláště v rozsáhlejších sítích.

3.5.2. Nastavení skenování

Pro skenování sítě jsou k dispozici různá nastavení, pomocí kterých si uživatel může přizpůsobit skenování dle potřeby. Možnosti jsou následující:

- Uživatel může omezit skenování na libovolný interval IP adres v rámci místní sítě. Výchozí hodnotou je skenování celé sítě. Pokud dojde ke změně sítě, meze intervalu se automaticky předvyplní na první a poslední možnou adresu v síti. V případě zadání adresy ve špatném formátu nebo adresy

mimo povolený interval se uživateli zobrazí chybová hláška a obnoví se poslední platná hodnota.

- Uživatel si může vybrat z různých metod skenování (podrobněji popsáno v kapitole číslo 2). Lze vybrat více metod najednou.
    - Basic scan (základní skenování) – výchozí metoda – v tomto režimu se provádí skenování pomocí technik: ICMP Echo scan, ICMP Timestamp scan, TCP SYN Ping scan portu 443 a TCP ACK Ping scan portu 80.
    - TCP SYN Ping scan uživatelem specifikovaného intervalu portů. Přesnější specifikace vstupních hodnot je popsána v kapitole 3.6.5.
    - TCP ACK Ping scan uživatelem specifikovaného intervalu portů
    - UDP Ping scan uživatelem specifikovaného intervalu portů
    - SCTP Init scan uživatelem specifikovaného intervalu portů
    - Skenování podle protokolů – uživatel zadá interval protokolů, jejich seznam lze nalézt na oficiálních stránkách IANA [13]
    - ICMP Mask scan
    - ARP Ping scan
- Uživatel může povolit nebo zamítnout skenování NetBios jména – toto skenování probíhá v jediném vlákně a v případě rozsáhlé sítě se touto možností může výrazně prodloužit doba skenování. Navíc si uživatel může nastavit dobu odezvy skenování NetBios jména pro jednotlivá zařízení v milisekundách.
- Uživatel může seřadit nalezená zařízení podle IP adresy, MAC adresy nebo podle poznámky a to buď ve vzestupném, nebo sestupném směru.
- Uživatel může nastavit dobu odezvy jednotlivých zařízení (*Host timeout*) a také čas, po kterém se skenování přeruší (*Scan timeout*). Tyto časy je třeba volit rozumně, jelikož příliš malé hodnoty mohou ovlivnit efektivitu skenování.

3.6. Analýza zařízení

Po výběru zařízení ze seznamu nalezených zařízení nebo po zadání IP adresy se dle výchozího nastavení spustí analýza zařízení pomocí metody TCP SYN. Zároveň dojde také k analýze síťového a NetBios jména a vypočítání průměrné odezvy. Po skončení analýzy se uživateli zpřístupní stránka s pěti záložkami, které popíši podrobněji níže.

3.6.1. Informace o zařízení

V záložce *Info* si může uživatel přečíst informace o daném zařízení. V případě skenování lokálního zařízení se jedná o IP a MAC adresu zařízení, síťové a NetBios jméno, název společnosti výrobce síťové karty zařízení, průměrnou dobu odezvy v sekundách (s přesností až na desítitisíciny sekundy) a také vlastní poznámku a přidělenou ikonu. Pokud se jedná o externí IP adresu, jsou možnosti omezeny pouze na IP adresu, síťové jméno a průměrnou odezvu.

Z této záložky lze rovněž pomocí tlačítek nastavit nebo změnit poznámku, vybrat novou ikonu, vymazat veškeré logy pro dané zařízení a také spustit nové skenování (vhodné zejména pokud uživatel změní nastavení). Změna poznámky probíhá pomocí dialogového okna, do kterého uživatel napíše novou poznámku (při vyvolání okna bude jako implicitní text původní poznámka, nebo „???“ v případě neznámého zařízení). Změna ikony probíhá podobně – uživatel si v dialogovém okně vybere požadovanou ikonu. Po kliknutí na ikonu bude změna ihned provedena. K dispozici je více než 160 různých ikon.

3.6.2. Síťové útoky

V této záložce může uživatel provádět různé síťové útoky na vybrané zařízení. K dispozici je deset útoků, některé z nich lze použít pouze na zařízení v místní síti (v opačném případě bude uživateli zobrazena chybová hláška a útok nebude proveden). Záložka se skládá ze dvou ovládacích prvků – *spinner* (neboli *combobox*) – z něj uživatel vybírá typ útoku – a tlačítko, kterým útok zahájí. Po zahájení útoku se zobrazí větší dialogové okno signalizující probíhající útok. V tomto okně je několik ovládacích prvků, pomocí kterých lze regulovat sílu útoku.

Prvním regulátorem lze dynamicky nastavovat počet odesílaných paketů za vteřinu. Maximální možná frekvence je 300000 paketů za sekundu, nicméně tato frekvence je pouze orientační a je závislá na maximální rychlosti přístupového bodu a také na zařízení. Dále je možno (pro testovací a experimentální účely) zapnout jemnější mód (checkbox *Low packet range*), ve kterém lze frekvenci přesně korigovat.

Druhý regulátor slouží k nastavení intervalu obnovení smyčky cyklu probíhajícího útoku. Na frekvenci útoků to nemá žádný vliv. Může být využit především k útokům s náhodou zdrojovou IP adresou nebo portem, protože nové náhodné hodnoty se generují pouze s každou iterací cyklu. I zde je zvolená hodnota pouze orientační, obzvlášť při nízkém intervalu, jelikož zařízení nemusí zvládat množství odchozích paketů a dochází tak ke zpoždění.

Posledním regulátorem lze měnit velikost odesílaných paketů. Paket obsahuje náhodná data o zvolené velikosti. Tento faktor výrazně ovlivňuje sílu útoku. U reflektivních útoků tato možnost není k dispozici, jelikož nemá význam.

V dialogovém okně se také nachází informace o odezvě daného zařízení. Tato informace je pouze orientační, protože i když je zařízení pod útokem, může odpovídat normálně (především u útoků, které způsobují odepření přístupu jiným způsobem, než zahlcením pakety). Nějaké výkyvy v konektivitě ovšem pozorovat lze. Uchovávány jsou vždy pouze poslední tři hodnoty.

Síťové útoky, které lze provádět, mohou být záplavového (*ICMP Echo flood*, *ICMP Destination Unreachable*, *ICMP Router Advertisement*, náhodné ICMP pakety, UDP pakety na port 0) nebo reflektivního (*ICMP Echo Amplification*, *UDP Amplification*, *DNS Amplification*) typu, dále útok *WinFreez* (přesměrování všech příchozích a odchozích paketů na vlastní IP adresu) a také útok *ARP Cache Poisoning*. Jednotlivé útoky jsou detailněji popsány ve druhé kapitole.

3.6.3. Seznam TCP a UDP portů

Tyto dvě záložky obsahují seznam otevřených portů na daném zařízení. Jednotlivé porty jsou charakterizovány číslem portu, které je zobrazeno velkým písmem, aby bylo dobře viditelné. Dále je pro každý port zobrazen zkrácený a úplný název služby. Databáze portů, které aplikace zná, je tvořena kombinací databáze nmap-services [1] a oficiální IANA databáze [13].

3.6.4. Logy

Poslední záložka obsahuje statistiku připojení pro dané zařízení. Pokaždé, kdy je dané zařízení objeveno skenováním, nebo je jeho IP adresa zadána přímo z hlavního menu, dojde k uložení záznamu o připojení. Každý záznam je charakterizován datem a časem připojení, externí IP adresou a názvem aktuálního přístupového bodu (SSID). Záznamy lze filtrovat podle data – uživatel může nastavit počáteční a koncové datum. Intervaly povolených dat jsou kontrolovány. V případě, že uživatel zadá nesprávný interval, objeví se chybová hláška a datum bude navráceno na poslední platnou hodnotu. Výchozím intervalem je statistika pro aktuální den.

3.6.5. Nastavení

V nastavení (*Port Scan Settings*) lze specifikovat intervaly portů, které mají být skenovány. Vybrány mohou být buď konkrétní porty, a/nebo interval portů definovaný pomlčkou. Jednotlivé porty nebo intervaly musí být od sebe odděleny čárkou. Příkladem vstupní hodnoty může být například: *21,23-27,80,400-500*.

Dále lze zvolit několik různých metod skenování portů, nejefektivnějšími metodami jsou TCP SYN a TCP Connect scan. Pro skenování UDP portů je k dispozici jediná možná metoda. Více informací o metodách skenování portů lze nalézt v kapitole 2.

Možnost Use System DNS Resolver umožňuje zjistit lokální síťové jméno za pomoci lokálního DNS serveru. Lze také specifikovat timeout pro skenování, po jehož uplynutí dojde k ukončení právě probíhající skenovací metody.

Lze také upravit nastavení pro útok DNS Amplification. Uživatel může specifikovat doménu, která bude použita k útoku (a tím pádem i nastavit sílu útoku), a také server, který bude vyhodnocovat podvržené DNS dotazy. Ve většině případů je vhodné použít lokální server kvůli nízké latenci.

3.7. Informace o síti

V této části aplikace lze nalézt veškeré informace o síti, ke které je zařízení aktuálně připojeno. Stránka je rozdělena do tří kategorií. První kategorie obsahuje informace o přístupovém bodu, externí IP adrese a o zemi, ve které je IP adresa registrována. Ve druhé kategorii jsou informace o místní síti – vnitřní IP adresa, adresa výchozí brány a *broadcastu*, maska podsítě a adresy DHCP a DNS serverů. Poslední kategorii tvoří jediná položka – MAC adresa zařízení.

4. Implementace

Tato kapitola je zaměřena na implementaci aplikace, popis jednotlivých tříd, objekty a jejich vzájemnou interakci. Celá aplikace byla vytvořena v nejnovější verzi vývojového prostředí Android Studio s využitím technik objektově orientovaného programování. Zdrojové kódy aplikace jsou napsány v jazyce Java Standard Edition verze 7. Implementace probíhala na operačním systému Windows 7.

4.1. Přehled aktivit a fragmentů

Aktivita (*Activity*) je druh třídy aplikací na platformě Android, která obsahuje pohledy (*View*) – grafické prvky. Aktivity jsou základním stavebním kamenem aplikací pro Android. Pro každou sekci aplikace je definována nová aktivita. Aktivity se mohou skládat z tzv. *Fragmentů*, což jsou v podstatě subaktivity, ke kterým se dá v rámci aktivity přistupovat například přes záložky. Aplikace obsahuje následující aktivity:

- `MainFrame` – hlavní aktivita, obsluhuje grafické prvky hlavního menu. Podle zvolené činnosti vytváří příslušná dialogová okna a volá další příslušné aktivity.
- `AboutAndHelp` – v této aktivitě je zobrazena nápověda aplikace, která je načtena z přiloženého jednoduchého *html* (soubor webové stránky) souboru.
- `NetworkInfoFrame` – zobrazuje textové informace o síti
- `ScanNetworkFrame` – má na starosti zobrazení ovládacích prvků pro skenování sítě a také výsledků skenování.
- `DeviceDetailsFrame` – tato aktivita se skládá z fragmentů a je pouze rozcestníkem. Také definuje záložky zobrazující jednotlivé fragmenty.
    - `DeviceDetailsInfoFragment` – zobrazuje textové informace o vybraném zařízení, dále obsahuje prvky pro nastavení poznámky a ikony a také tlačítko umožňující spuštění nové analýzy zařízení
    - `DeviceDetailsTCPFragment`, `DeviceDetailsUDPFragment` – zobrazují seznamy nalezených portů

    - `DeviceDetailsLogsFragment` – obsahuje statistiku připojení zvoleného zařízení a také ovládací prvky a implementaci filtrovacích prvků.

- Aplikace dále obsahuje třídy typu *Preferences* definující nastavení pro jednotlivé sekce aplikace.

4.2. Android manifest

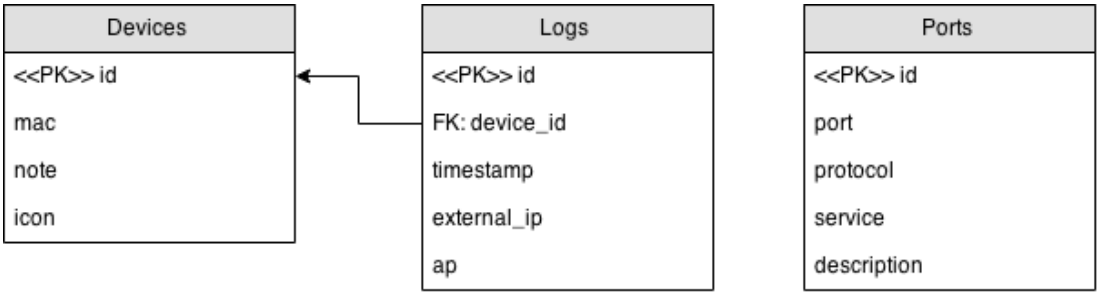
Jedná se o soubor definující hlavní parametry aplikace jako je seznam aktivit a jejich parametrů, stylů a požadovaných přístupových oprávnění. Aplikace vyžaduje oprávnění pro přístup k WiFi a internetu.

4.3. Po spuštění

Po spuštění aplikace dojde k načtení hlavní aktivity. Vytvoří se instance třídy `Singleton` – jedná se o třídu návrhového vzoru jedináček. Tato třída je určena k vzájemnému propojení všech aktivit a slouží jako globální rozhraní celé aplikace. Obsahuje důležité komponenty nezbytné pro běh aplikace, například globální kontext (lze chápat jako globální prostředí), výsledky posledního skenování, poslední nalezené zařízení, odkaz na právě spuštěnou aktivitu, přístup k databázi atd.

Jelikož načtení aplikace je poměrně zdlouhavý proces, jsou náročnější úkony spuštěny v novém vlákně pomocí *AsyncTask* (činnost prováděná na pozadí s podporou přímé aktualizace prvků uživatelského rozhraní). Načítání obstarává třída `BackgroundLoad`. Během načítání dochází k vytvoření *BroadcastReceiveru*, který provádí akce při změně připojení. Zajistí načtení potřebných údajů z nové sítě, aby mohla aplikace dále fungovat i po změně připojení.

Posledním krokem je načtení databáze. Správu databáze a veškeré akce spojené s přístupem k databázi zajišťuje třída `SQLManager` a pomocné třídy `SQLDeviceResult`, `SQLPortResult` a `SQLLogResult`, které charakterizují výsledky vyhledávání v databázi a zajišťují snadnější práci s nalezenými daty. Pokud databáze ještě nebyla vytvořena, aplikace vytvoří novou databázi s tabulkami dle schématu 4.1. Také naplní tabulku portů daty. Vzhledem k tomu, že je seznam portů poměrně rozsáhlý, bylo nutné vkládání portů při prvním spuštění optimalizovat. Nejlepší možností bylo sloučení všech databázových transakcí do jedné a tím minimalizovat přístup k databázi. Provedená optimalizace snížila čas tvorby tabulky portů přibližně dvěstěkrát.



Obrázek 4.1: Schéma databáze

4.4. Práva root uživatele

Aplikace ke svému chodu vyžaduje přístupová práva root uživatele. Jsou potřeba pro spouštění externích binárních souborů a také pro širší možnosti skenování sítě a síťových útoků. Operační systém Android totiž bez těchto práv neumožňuje uživateli pracovat na nižší než aplikační vrstvě modelu sítě (neumožňuje odesílat vlastní pakety).

Bezprostředně po spuštění aplikace odešle příkaz na linuxové jádro ve tvaru `su`. Poté čeká na uživatelovu odpověď. Komunikace probíhá prostřednictvím externích knihoven RootTools a RootShell. Tyto nástroje zajišťují komunikaci s knihovnami Nmap a Nping, které jsou jádrem celé aplikace. Správu celé této komunikace mají na starost třídy `BinaryHandler` a `BinaryCommandWrapper`.

4.5. Skenování sítě

Po kliknutí na tlačítko *Start scan* se spustí *AsyncTask* `BackgroundNetworkScan`. Nejprve se vymaže stávající seznam výsledků posledního skenování a poté dojde ke kontrole platnosti údajů v nastavení aplikace. V případě chyby bude uživateli zobrazena chybová hláška a proces se ukončí. Dále dojde k zavolání jednotlivých příkazů pro zvolené skenovací metody. Příkazy jsou ve formátu:

```
nmap <method> [<method ports>] --host-timeout <ht>
--max-rtt-timeout <ht> <targets>
```

kde:

- <method> – zkratka zvolené skenovací metody (podrobněji popsáno ve druhé kapitole)
- <method ports> – intervaly portů (pokud je zvolená metoda vyžaduje)
- <ht> – uživatelem specifikovaný timeout pro jednotlivá zařízení

- <targets> – interval skenovaných IP adres ve formátu například 192.168.0.5 192.168.1.10-* 192.168.2.*. Převod formátu intervalu z nastavení na formát pro knihovnu Nmap zajišťuje funkce `convertIpRange`. Pokud je zvoleno skenování celé sítě, je cíl zapsán pomocí CIDR notace, například takto: 192.168.1.0/23.

Po dokončení každé metody je výsledek předán třídě `NmapParser`, která z textového výstupu knihovny Nmap získá seznam IP adres nalezených zařízení a přidá jej do nového dočasného seznamu zařízení. Tento seznam je sloučen s výsledným seznamem a jsou odstraněny duplicitní nálezy. Jedná se o seznam objektů třídy `ScanResultItem` reprezentující jednotlivá zařízení. Každý prvek seznamu obsahuje informace o zařízení (IP a MAC adresu) a lze do něj také vložit poznámku a ikonu. Pro zobrazení seznamu uživateli je použita vlastní třída `ScanResultItemAdapter`.

Po skončení všech probíhajících skenovacích metod nastává ukončovací fáze. Nejprve se nastaví výchozí poznámky a ikony. Pro výchozí bránu a vlastní zařízení jsou nastaveny speciální hodnoty, aby tato zařízení uživatel v seznamu snadno rozpoznal. Aplikace poté vyhledá NetBios jméno každého zařízení (pokud je tato možnost zapnuta). To je zprostředkováno knihovnou JCIFS. V případě nálezu se nastaví NetBios jméno jako poznámka daného zařízení. Poté je pro každé zařízení vložen log a je vyhledána poznámka a ikona z databáze. Dojde k přepsání implicitních hodnot.

V případě násilného ukončení skenování (stiskem tlatíčka *Stop scan*) aplikace ukončí všechny běžící procesy s názvem `nmap` a provede finalizaci. Tím je zajištěno okamžité ukončení. NetBios jména zažízení v tomto případě nebudou vyhledávána.

4.6. Analýza zařízení a skenování portů

Po kliknutí na nalezené zařízení se automaticky spustí jeho analýza dle nastavení. Vytvoří se nový objekt `DeviceDetailsResult` a naplní se již zjištěnými informacemi. V případě ručního zadání IP adresy se do objektu vloží pouze IP adresa a ostatní informace jsou doplněny před skenováním portů. Množství zjišťovaných informací závisí na tom, jestli patří IP adresa do místní sítě či nikoliv.

Skenovací proces je spouštěn stejným způsobem jako u skenování sítě. Proces však nelze přerušit a uživateli je prezentován jako *ProgressDialog*. Pokud dojde

během procesu k odpojení od sítě, proces se sám ukončí až po uplynutí nastaveného timeoutu.

Nejprve dojde k ověření platnosti zadaných portů a poté se postupně volají příkazy pro zvolené metody. Formát příkazu a zpracování výsledků je podobný jako u skenování sítě.

Po dokončení skenování se *ProgressDialog* ukončí a uživateli se zpřístupní jednotlivé záložky. Výsledky skenování jsou uloženy v objektu reprezentujícím zařízení. Seznam nalezených portů je reprezentován objektem `PortResultItem` a jeho zobrazení zajišťuje objekt třídy `PortResultItemAdapter`. TCP a UDP porty jsou od sebe odlišeny vlastním seznamem a záložkou.

4.7. Síťové útoky

Po vybrání požadovaného útok a kliknutí na tlačítko se spustí *AlertDialog* a zároveň s ním nová vlákna tříd `Flood` a `Ping`.

Třída `Ping` pravidelně posílá příkaz ve tvaru `ping -c 1 <target>`, kde `<target>` je IP adresa cílového zařízení. Získané výsledky jsou zpracovány a předány dialogovému oknu. Vždy po získání nového výsledku je třetí nejstarší výsledek vymazán.

Třída `Flood` (z angličtiny *flood* = záplava) má na starosti průběh útoku. Útoky jsou realizovány pomocí nekonečné smyčky příkazů. Každý útok je reprezentován pravidelným odesíláním vhodně sestavených paketů. Tuto činnost zajišťuje knihovna Nping.

Některé útoky realizovat pouze v rámci místní sítě. V případě, že se uživatel pokusí provést nevhodný typ útoku na zařízení mimo síť, se útok ani nespustí a uživatel bude informován chybovou hláškou.

Přerušení útoku je reprezentováno voláním příkazu `killall nping`, který probíhající útok ihned spolehlivě ukončí. Při ukončení útoku dojde také k ukončení činnosti vlákna `Ping`.

Následuje implementace jednotlivých útoků. Zde je vysvětlení jednotlivých zkratek použitých v příkazech:

- `<rate>` značí nastavený počet odesílaných paketů za sekundu
- `<refresh>` symbolizuje desetinné číslo vyjadřující počet vteřin. Pro docílení požadovaného efektu je tato hodnota násobena hodnotou `<rate>`. Výsledek násobení je zaokrouhlen na celé číslo.
- `<size>` znamená navýšení velikosti paketu o danou hodnotu
- `<target>` - IP adresa zařízení

- <gateway> - IP adresa brány
- <broadcast> - IP adresa *broadcastu*

Popis některých parametrů:

- `--rate` – počet odeslaných paketů za sekundu
- `-c` – celkový počet odeslaných paketů
- `-S` – zdrojová IP adresa
- `--source-port` – zdrojový port
- `-p` – cílový port
- `--icmp`..., `--arp`..., `--udp` – použitý protokol a jeho upřesňující vlastnosti

4.7.1. Záplavové a reflektivní útoky

Implementace těchto útoků je snadná, jelikož se jedná o jediný příkaz. Zmíním zde pouze formát odesílaného příkazu, princip útoků je popsán ve druhé kapitole.

- **ICMP Echo Flood**: `nping --icmp --icmp-type echo -S random --rate <rate> -c <rate>*<refresh> --data-length <size> -q <target>`
- „**Smurf**“: `nping --icmp --icmp-type echo -S <target> --rate <rate> -c <rate>*<refresh> -q <broadcast>`
- „**Fraggle**“: `nping --udp -p 7 -S <target> --rate <rate> -c <rate>*<refresh> -q <broadcast>`
- „**Smack**“: `nping --icmp --icmp-type 3 --icmp-code random -S random --rate <rate> -c <rate>*<refresh> --data-length <size> -q <target>`
- „**Twinge**“: `nping --icmp --icmp-type random -S random --rate <rate> -c <rate>*<refresh> --data-length <size> -q <target>`
- „**P-Smash**“: `nping --icmp --icmp-type -S random --rate <rate> -c <rate>*<refresh> --data-length <size> -q <target>`
- „**WinFreez**“: `nping --icmp --icmp-type redirect --icmp-code net --icmp-redirect-addr <target> -S random --rate <rate> -c <rate>*<refresh> -q <target>`

- **UDP Zero flood**: `nping --udp -p 0 --rate <rate> -c <rate>*<refresh> --data-length <size> -q <target>`

4.7.2. ARP Cache Poisoning

Tento útok se od ostatních odlišuje tím, že nelze ovládat jeho síla. Má pevně stanovené hodnoty `<rate>` a `<refresh>`. Dále je k útoku potřeba znát MAC adresu brány, což je zajištěno funkcí `getMacFromArpCache`. Útok se skládá ze dvou příkazů, jeden je odesílán bráně a druhý oběti.

```
nping --arp-type arp-reply --arp-sender-mac <myMac> --arp-sender-ip <gatewayIp> --arp-target-mac <targetMac> --arp-target-ip <targetIp> -- rate 20 -c 100 <targetIp>
```

```
nping --arp-type arp-reply --arp-sender-mac <myMac> --arp-sender-ip <targetIp> --arp-target-mac <gatewayMac> --arp-target-ip <gatewayIp> -- rate 20 -c 100 <gatewayIp>
```

4.7.3. DNS Amplification Attack

Implementace tohoto útoku je náročnější. Je potřeba definovat přesnou strukturu paketu. Knihovna Nping touto možností disponuje, ale je potřeba data paketu zapsat v hexadecimálním tvaru. Ke konstrukci částí paketu slouží metody třídy `HexTools`. Přesnou strukturu paketu lze nalézt v kapitole 2. ID transakce je generováno náhodně pomocí funkce `randomNDigitHex(4)` a korektní převod domény do hexadecimálního tvaru zajišťuje funkce `encodeHexUrl`. Je zde však omezení – jednotlivé části domény nesmí být delší než 256 znaků a nesmí obsahovat znaky mimo základní ASCII tabulku. V takovém případě bude během převodu vyvolána výjimka, které je zpracována ve formě chybové hlášky zobrazené uživateli.

Finální příkaz pro tento útok vypadá následovně (<DNSIP> označuje IP adresu zvoleného DNS serveru):

```
nping --udp --source-port random -p 53 -S <target> --rate <rate> -c <rate>*<refresh> --data <packet> -q <DNSIP>
```

Aplikace je schopna interpretovat tak silný útok, že je omezen pouze průtokem dat přístupového bodu, odezvou DNS serveru a specifikací domény.

4.8. Logy

Logy jsou reprezentovány záznamy v databázi. Časový údaj je uložen jako *timestamp*, tedy celočíselný zápis informace o datu a čase s přesností na tisíciny vteřiny.

Filtrování záznamů realizují dva *DatePickerDialogy* umožňující snadný a rychlý výběr data. Po nastavení požadovaného intervalu dojde k vyhledání logů s datem v daném intervalu v databázi.

4.9. Informace o síti a zpracování IP adresy

Načtení informací o síti zajišťuje třída `WifiReceiver`. Po detekci nového připojení volá metodu `refresh` třídy `NetInfo`. Objekt této třídy je navrhnut jako jedináček a nese informace o aktuální síti. Při získávání IP adres ze zařízení byl nutný jejich převod, jelikož byly obdrženy v celočíselném formátu a také s prohozenými oktety. Převod IP adresy na řetězec realizuje funkce `convertRetrievedIp` třídy `NetInfo` a je řešen následujícím algoritmem:

- Nejprve dojde k převodu na datový typ long, jelikož IP adresa je 32bitová hodnota, stejně jako datový typ Integer, ve kterém je získána, tím pádem může být i ve tvaru záporného čísla. Převod je vyřešen bitovým součtem: `long ip = ipAddress & 0x00000000FFFFFFFFL;`
- Poté jsou bitovým posunem získány jednotlivé oktety:
    - `octet1 = ip & 0xFF;`
    - `octet2 = (ip & 0xFF00) >>> 8;`
    - `octet3 = (ip & 0xFF0000) >>> 16;`
    - `octet4 = (ip & 0xFF000000) >>> 24;`
- Výsledná IP adresa je dána konkatenací jednotlivých oktetů.

IP adresa je reprezentována jako řetězec. Aby však mohly být ověřovány intervaly IP adres, bylo nutné implementovat převodní funkce na datový typ `long` a zpět. Příslušné metody `ipToLong` a `longToIp` jsou obsaženy v třídě `NetInfo`.

Aplikace také potřebuje znát první a poslední IP adresu v síti. Tyto funkce bylo rovněž nutno implementovat. První adresa je vypočítána následovně:

```
startIp = hostIp – hostIp % (ipToLong (getWildcardFromMask (subnetMask)) + 1);
```

kde `hostIp` je aktuální IP adresa zařízení. Metoda `getWildcardFromMask` invertuje masku podsítě a převodem této „IP adresy“ na celé číslo lze získat počet

možných adres v síti. Pro získání koncové IP adresy stačí k počáteční adrese tuto hodnotu přičíst.

Externí IP adresa je získána pomocí připojení k webové stránce [14] s potřebnými údaji. Ty jsou poté zpracovány. Během testování jsem zjistil, že v některých případech přijde poškozená odpověď. Proto se tento proces opakuje až pětkrát. Pokud není adresa získána ani posledním pokusem, je hodnota nahrazena textem *Connection error*.

4.10. Nastavení

Pro tvorbu nastavení aplikace byly využity speciální aktivity typu *PreferenceActivity*, umožňující zpracovat XML soubory s jednotlivými prvky nastavení. Nastavení je poté řešeno pomocí *SharedPreferences*. Jedná se o perzistentní objekt, do kterého lze ukládat a načítat z něj základní datové typy (*boolean*, *integer*, *long*, *float*, *string*).

Pro komunikaci s objektem *SharedPreferences* byla vytvořena třída `Opt`, která z nastavení načítá jednotlivé hodnoty.

Některé hodnoty nastavení je potřeba před uložením ověřit (např. korektní IP adresa, interval portů atp.). K tomuto účelu jsou v jednotlivých aktivitách implementovány validační metody, které před zápisem nové hodnoty ověří jejich správnost. V případě chyby nebude nová hodnota zapsána a uživateli se zobrazí chybová hláška.

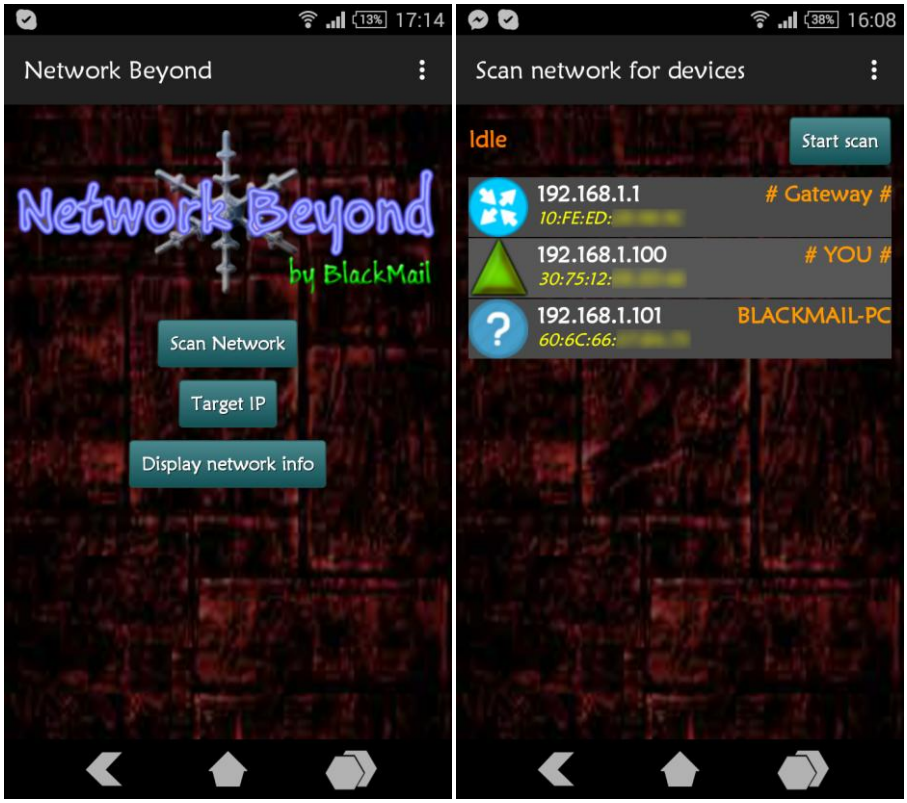
4.11. Tvorba GUI

GUI neboli grafické uživatelské rozhraní (z angličtiny *Graphical User Interface*) je při vývoji Android aplikací reprezentováno XML soubory definující jeho přesnou podobu. Pro každou aktivitu, seznam a také pro dialogové okno útoku bylo potřeba vytvořit samostatný XML soubor.

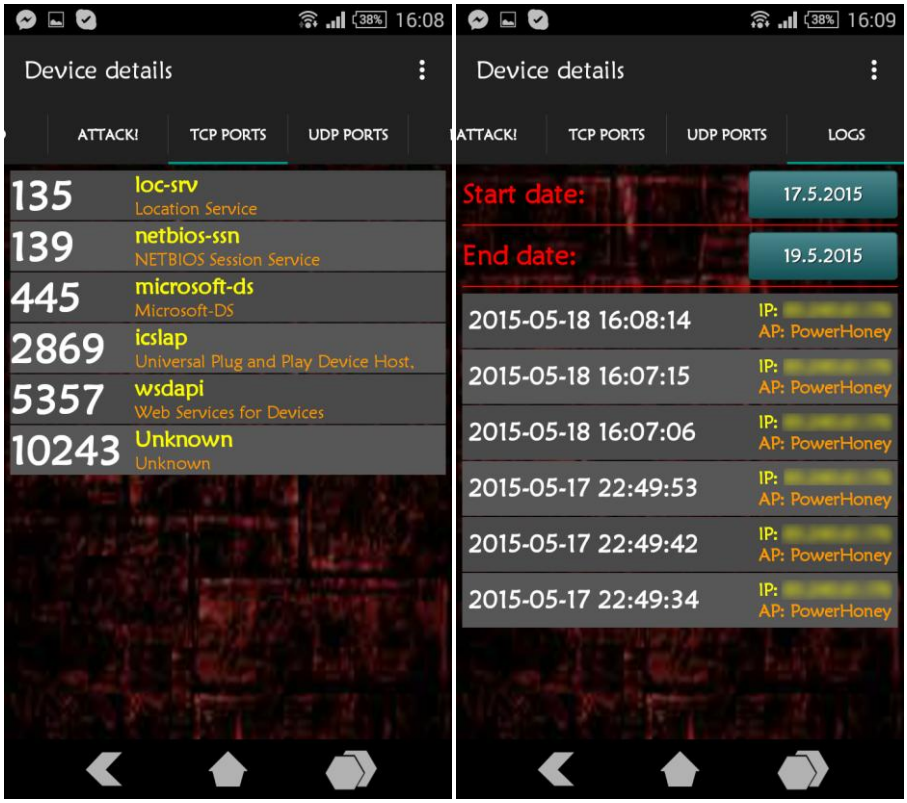
GUI lze psát buď ručně, nebo lze využít vestavěný vizuální editor Android Studia, díky kterému lze snadno sestavit. Při vývoji této aplikace jsem používal pouze vestavěný editor. K dispozici je velké množství různých ovládacích prvků. Jednotlivé prvky GUI jsou umístěny za pomocí *Layoutů*, tím pádem GUI vypadá stejně na zařízeních s různým rozlišením obrazovky.

Mimo klasických prvků GUI aplikace obsahuje také vlastní typ `FlowLayout`, který dovoluje zobrazit předem neznámý počet menší prvků s využitím zalamováním řádků. Této vlastnosti je využito například při výběru ikony zařízení.

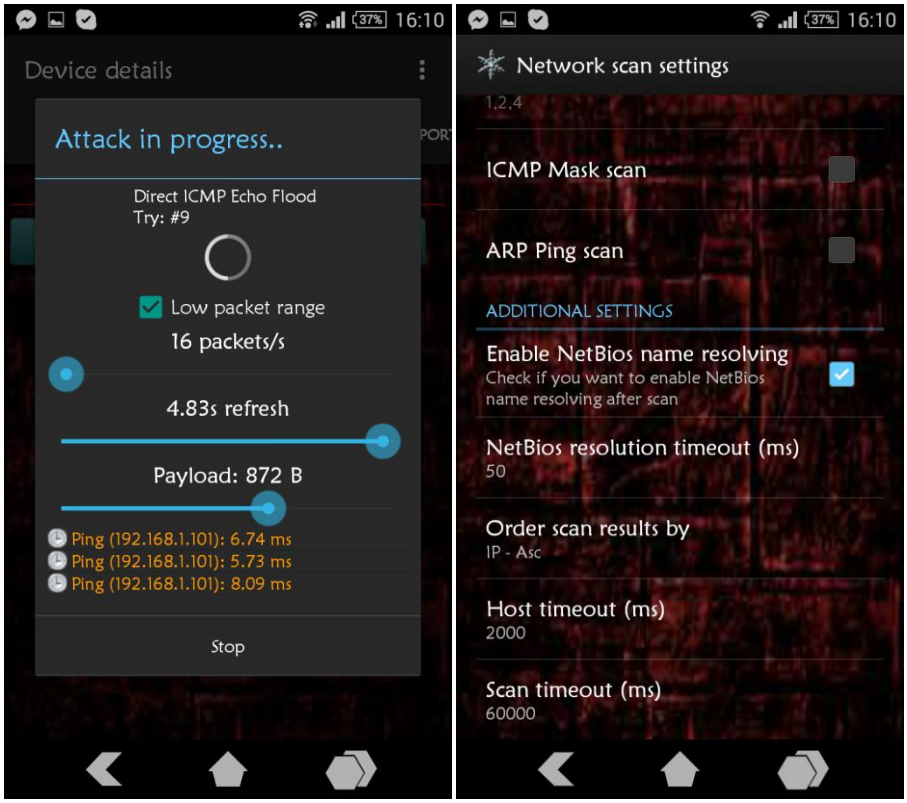
Aplikace také obsahuje vlastní font *Maiandra GD* společnosti *Microsoft*. Změnu fontu v celé aplikaci zajišťuje třída `FontsOverride`. Následuje ukázka GUI aplikace.



Obrázky 4.2 a 4.3: Hlavní menu aplikace a výsledek skenování sítě



Obrázky 4.4 a 4.5: Seznam otevřených portů a ukázka logů.



Obrázky 4.6 a 4.7: Probíhající útok a nastavení

5. Testování a vyhodnocení

5.1. Testování

Aplikace *Network Beyond* byla vyvíjena a testována na zařízení *Sony Xperia Z1 Compact* s operačním systémem Android 4.4.4. Při vývoji byl také kladen ohled na jiná možná rozlišení displeje – Android Studio umožňuje zobrazit náhled GUI pro libovolné rozlišení. Nicméně díky využití vizuálního editoru a dostatečně vysokému rozlišení obrázků nenastaly s přenosem na jiná rozlišení žádné potíže.

Aplikace byla testována na domácí síti, při vývoji útoků byl použit program na zachytávání paketů Wireshark. Testováno bylo rovněž několik dalších domácích sítí a také obě školní sítě VUT. Díky testování na velkých sítích bylo odhaleno a opraveno spoustu drobných chyb a aplikace byla optimalizován tak, aby skenování v takto rozlehlých sítích probíhalo dostatečně rychle. Oběťmi byla různá zařízení – stolní počítače, notebooky i mobilní telefony. Na všech testovaných sítích a zařízeních se skenování a provádění útoků zdařilo.

Během vývoje jsem prezentoval aplikaci svým známým a ptal se na jejich názory ohledně vzhledu aplikace. Díky jejich zpětné vazbě dosáhla aplikace své finální podoby.

5.2. Vyhodnocení výsledků aplikace

Skenování sítě probíhá velmi rychle. V domácí síti trvá běžné skenování asi tři vteřiny, na větších sítích však až řádově desítky sekund. Dobu skenování výrazně ovlivňuje nastavení aplikace.

Síla jednotlivých útoků se mírně lišila dle sítě. V domácí síti fungovala většina prováděných útoků bezproblémově. Nejsilnějším útokem je ARP Cache Poisoning, který zařízení odpojil od sítě okamžitě. Dalším silným útokem je jednoduchý ICMP Echo Flood, který dokázal zahltit cílové zařízení během chvilky. Útok DNS Amplification dosahoval pozoruhodných výsledků, avšak při testování jsem neměl k dispozici vhodnou doménu, abych dosáhl maximální možné amplifikace. Nicméně i s koeficientem amplifikace přibližně 9 oběť nemohla zobrazovat žádné webové stránky.

6. Závěr

Cílem této práce bylo vytvořit aplikaci na skenování sítě, která je schopna provádět také síťové útoky. Výsledkem implementace je aplikace s názvem *Network Beyond*.

Aplikace poskytuje uživateli propracovaný síťový skener s rozsáhlým nastavením. Při vývoji byl kladen důraz na jednoduchost, snadné ovládání a zároveň také na přehlednost a zpracování velkého množství informací. Seznamy bylo třeba navrhnout tak, aby se v nich uživatel snadno orientoval a rychle nalezl nejpodstatnější informace. Při vývoji síťových útoků jsem se zaměřil na možnost grafické reprezentace a dynamického nastavení síly útoku.

Aplikace by se dala více rozšířit, například přidáním dalších útoků, možnosti exportu dat, terminálu pro přístup k jednotlivým otevřeným službám, vyhledávání jména *Bonjour*, možností rozšířených logů, automatickým nebo plánovaným skenováním atd. Také by se dalo eliminovat použití knihovny Nmap a tím pádem poskytovat výsledky již v průběhu skenování.

Při vývoji aplikace jsem se seznámil s principem implementace aplikací pro Android, detailně pochopil principy a implementaci síťových útoků, naučil se kombinovat knihovny pro různé platformy.

Literatura

[1] Lyon, G. F.: *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. NMAP PROJECT, 2009, ISBN: 978-0-9799587-1-7

[2] Postel, J.: Transmission Control Protocol, 1981
URL https://tools.ietf.org/html/rfc793

[3] Přispěvatelé Wikipedie, Ping flood [online], Wikipedie: Otevřená encyklopedie, © 2015, Datum poslední revize 30. 03. 2015, 07:52 UTC
URL http://cs.wikipedia.org/w/index.php?title=Ping_flood&oldid=12416060

[4] Přispěvatelé Wikipedie, Smurf attack [online], Wikipedie: Otevřená encyklopedie, © 2014, Datum poslední revize 12. 12. 2014, 07:42 UTC
URL http://cs.wikipedia.org/w/index.php?title=Smurf_attack&oldid=12068716

[5] Mirkovic, J.: DDoS Attack List
URL http://www.isi.edu/~mirkovic/bench/attacks.html

[6] Arboi, M., 2002
URL http://www.securityspace.com/smysecure/catid.html?id=11024

[7] Sanders, Ch. Understanding Man-in-the-Middle Attacks – ARP Cache Poisoning (Part 1), 2010
URL
http://www.windowsecurity.com/articles-tutorials/authentication_and_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part1.html

[8] Mislove, A.: Project 1 Primer: DNS Overview, 2009
URL
http://www.ccs.neu.edu/home/amislove/teaching/cs4700/fall09/handouts/project1-primer.pdf

[9] Prince, M., 2012
URL https://blog.cloudflare.com/deep-inside-a-dns-amplification-ddos-attack/

[10] Rozekrans, T., de Koning, J.: *Defending against DNS reflection amplification attacks*, 2013
URL
https://www.nlnetlabs.nl/downloads/publications/report-rrl-dekoning-rozekrans.pdf

[11] Cornell, D.: DNS Amplification Attacks, 2014
URL https://labs.opendns.com/2014/03/17/dns-amplification-attacks/

[12] Mockapetris P.: Domain names – Implementation and specification, 1987
URL http://tools.ietf.org/html/rfc1035

[13] Internet Assigned Numbers Authority (IANA), 1995
URL http://www.sockets.com/services.htm

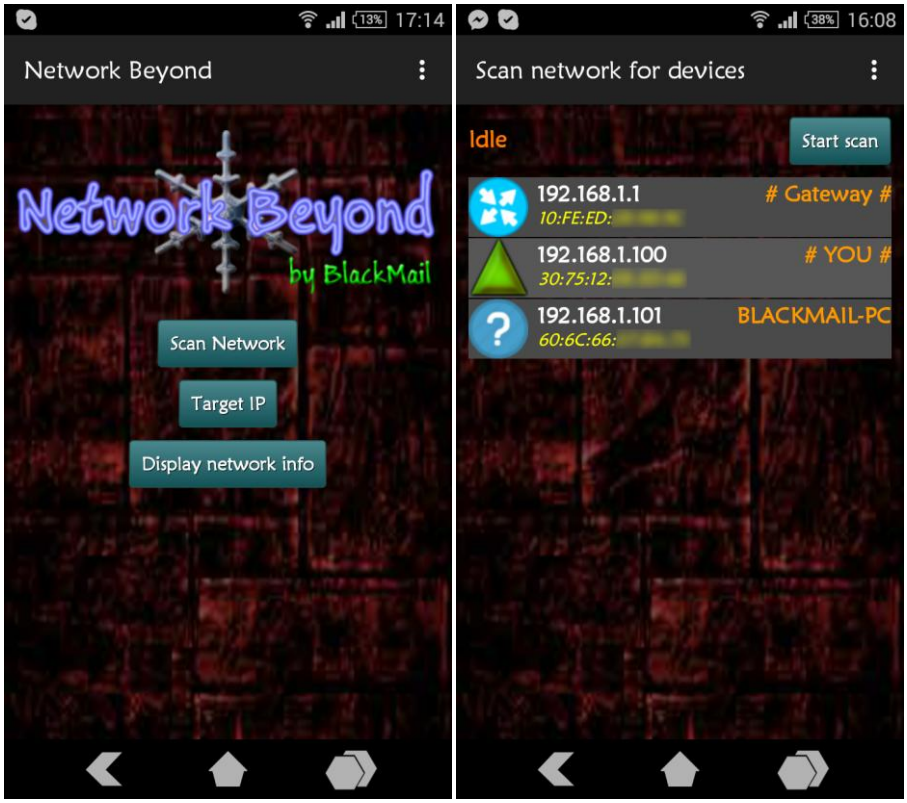
[14] URL http://ip2country.sourceforge.net/ip2c.php?format=JSON

A. Obsah CD

javadoc	JavaDoc dokumentace ke zdrojovému kódu
licence	Složka s licencemi jednotlivých knihoven
source	Složka se zdrojovými kódy a knihovnami aplikace
thesis	Složka obsahující text práce
networkbeyond.apk	Instalační soubor aplikace *Network Beyond*
readme.pdf	Návod k ovládání aplikace

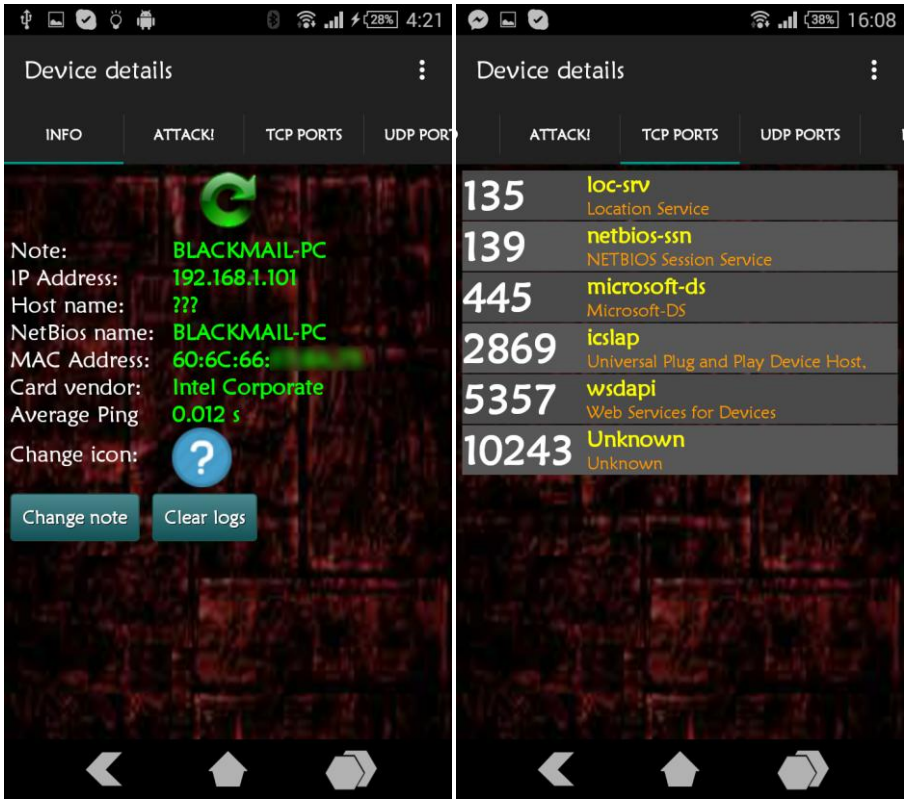
B. Návod k ovládání aplikace Network Beyond

Po spuštění aplikace dojde k vyžádání root práv a po krátké instalaci aplikace přejde do hlavního menu.

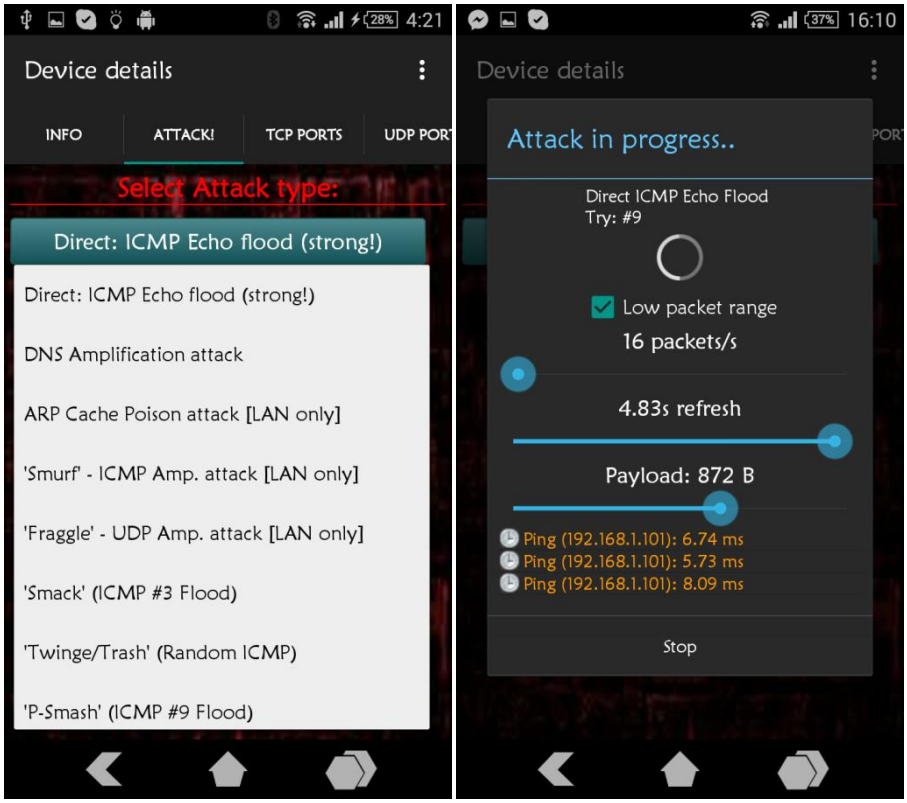


Obrázky B.1 a B.2: Hlavní menu a skenování sítě

Po kliknutí na tlačítko *Scan Network* se zobrazí seznam zařízení z posledního skenování. Tlačítko *Start Scan* zahájí skenování. Po kliknutí na nalezené zařízení se spustí jeho analýza, kterou lze rovněž vyvolat z hlavní nabídky kliknutím na *Target IP* a zadáním IP adresy. Po dokončení analýzy lze kliknutím na záložku nebo přetažením záložky vybrat požadovanou sekci. V sekci *Info* lze měnit také ikonu a poznámku, vymazat logy nebo spustit novou analýzu (zelená šipka). Nastavení zobrazení dat v záložce *Logs* lze provést kliknutím na *Start Date*, resp. *Stop Date*. Výběr útoku je realizován kliknutím na tlačítko s názvem útoku a následným výběrem ze zobrazené nabídky. Intenzitu útoků lze nastavovat posuvníky. Nastavení a nápovědu lze vyvolat odkudkoliv kliknutím na tři tečky v pravém horním rohu.



Obrázky B.3 a B.4: Informace o zařízení, záložka s otevřenými porty



Obrázky B.5 a B.6: Výběr útoku a probíhající útok.